

Zadanie

Naprogramujte REST API v programovacom jazyku Python 3.10+ podľa poskytnutej Dokumentácie v nástroji Swagger (link: [PES APIv1](#), verzia dokumentácie: 1.2.0 alebo vyššia).

Prosím prečítajte si pozorne „Poznámky k Dokumentácii a Databáze“, obsahuje dôležité informácie a zmeny oproti Swagger dokumentácii.

Kontext úlohy:

API je súčasťou webovej aplikácie na evidenciu a správu pôžičiek Fyzickým osobám. Aktérm konkrétnej pôžicky sú maximálne 3 osoby: Klient/Dlžník, Ručiteľ a Referent (pre potreby tohto zadania zastupuje Veriteľa v plnej mieri, plní rolu správcu systému a poskytovateľa služby). API poskytuje základné informácie o pôžičkách a zúčastnených stranach (Dlžník a Ručiteľ) externej Webovej službe – Portál Elektronických Služieb (PES), prostredníctvom ktorej si môže Klient zobraziť niektoré informácie o svojich pôžičkách, nahrávať dokumenty, upravovať kontaktné údaje Ručiteľa a seba samého.

Zdroj dát:

Ako zdroj dát použite PostgreSQL databázu (priložený súbor **cmdb_a2-part.sql** – PostgreSQL DB Dump). Databáza obsahuje finálnu zostavu tabuľiek a nie je dovolené vytvárať nové tabuľky. V prípade potreby máte povolené vytvárať vlastné databázové View a Funkcie a zasahovať do záznamov v databáze.

Základné tabuľky potrebné pre splnenie úlohy “public.loan” (endpoinky /loan*) a “public.natural_person” (endpoinky /contact*; natural_person = Fyzická osoba, Klient/Dlžník/Ručiteľ/Referent/...) obsahujú ukážkové dáta.

V prípade, že nedokážete z poskytých dát alebo databázovej štruktúry vysvetliť náväznosť na konkrétny Field v dátovej schéme v Dokumentácii, je možné tento Field vyplniť napr. prázdnou hodnotou, stringom “N/A” atď.

Forma výstupu:

Riešenie zadania odovzdáte vo forme Docker kontaineru/ov a voliteľne konfigurácie orchestrátoru Docker-Compose. Riešenie musí obsahovať zdrojový kód, Dockerfile a README s pokynmi pre build a spustenie pre každý kontainer. V prípade použitia orchestrátoru sú tiež povinné príslušné konfiguračné súbory a priložené pokyny v README. Príklad výstupu, názvy ani rozsah nie sú záväzné (riešenie môže byť aj jeden kontainer):

```
pes_apiv1.zip:  
|--- pes_apiv1_container1/  
|   |--- src/  
|   |--- Dockerfile  
|   |--- README  
|--- pes_apiv1_container2/  
|   |--- src/
```

```
| ...
|--- docker_compose_conf.yml
|--- README
```

...

Riešenie musí byť vo forme standalone aplikácie/služby, ktorá má jasne definovaný postup ako aplikáciu nainštalovať/nakonfigurovať a spustiť. Povolené, resp. očakávané závislosti (dependencies) riešenia sú: aktuálna distribúcia Linux OS (napr. Ubuntu 24.04, Fedora 41, atd.), nainštalovaný Docker a Docker Compose. V prípade, že Vaše riešenie bude vyžadovať ďalšie závislosti alebo prerekvizity musí byť súčasťou riešenia inštalačný/konfiguračný skript a príslušné pokyny v README.

Technické požiadavky na riešenie:

REST API bude sprístupnené na porte 80 na IP adrese/localhoste stroja, na ktorom bude spustené. Príklad: <http://127.0.0.1/pes/apiv1/loans>

Riešenie bude možné vybuildovať a spustiť na Ubuntu 24.04, Fedora 41 alebo inej príbuznej Deb/RHEL distribúcii

Riešenie bude obsahovať poskytnutú PostgreSQL databázu so zdrojovými dátami. Databáza môže byť súčasťou kontaineru, ktorý bude obsahovať REST API alebo môže poskytnutá ako standalone kontainer (v tom prípade ale musí byť zahrnutá do orchestrátoru)

Zdrojový kód bude písaný v angličtine. Python súbory budú formátované defaulte, tj:

Encoding: UTF-8, Indent: 4 spaces

Postup testovania:

Vaše riešenie bude prvý krát otestované bez Vašej prítomnosti a možnosti poskytnúť ďalšiu nápovedu alebo dodatočné pokyny. Druhé testovanie prebehne v prípade potreby na online/osobnom pohovore s možnosťou Vásheho vstupu do procesu testovania. Testovanie bude vždy prebiehať od začiatku, tzn. inštaláciou Vašeho riešenia na virtuálny stroj s čistým OS a nainštalovaným Dockerom a DockerCompose. Súčasťou druhého testovania môže byť krátky pohovor a analýza Vašeho zdrojového kódu a spôsobu riešenia zadania.

Samotný postup testovania:

1. Inštalácia Vašeho riešenia na čistý OS vo virtuále
2. postupné testovanie implementovaných endpointov z Dokumentácie
3. analýza zdrojového kódu

Hodnotené kritéria:

1. celková funkčnosť riešenia – riešenie je inštalovateľné a spustiteľné podľa dodaných inštrukcií a plní aspoň čiastočne funkcionality popísanú v Dokumentácii
2. kvalita zdrojového kódu
3. schopnosť vysvetliť/zdôvodniť použité postupy v prípade, že pri analýze Vašeho riešenia vzniknú otázky na túto tému

Podmienkou splnenia zadania nie je kompletnosť riešenia – tzn. nie je nutné implementovať všetky endpointy z poskytnutej Dokumentácie.

Zdrojový kód nemusí obsahovať komentáre a žiadnu formu testov.

Poznámky k Dokumentácii a Databáze:

Väčšina endpointov obsahuje povinnú kombináciu parametrov “eduid”, “first_name”, “last_name”, “person_id” a “birth_date”. Jedná sa o chybu/nepresnosť – v skutočnosti je povinný buď parameter “eduid” alebo kombinácia parametrov “first_name” + “last_name” + “person_id” + “birth_date”.

Parameter “person_id” je Rodné číslo “public.natural_person.personal_identification_number”). Parameter “eduid” je unikátny identifikátor v našom systéme. Tieto údaje nájdete v databáze v tabuľkách “public.natural_person” a “public.natural_person_attribute”, resp. v príslušných Views.

Pre potreby tohto zadania bude stačiť implementovať iba niektoré API endpointy, ideálne s použitím všetkých parametrov (zvolené API endpointy prosím popíšte v README).

Príklad URL: <http://127.0.0.1/pes/apiv1/loans?eduid=123456>

alebo

http://127.0.0.1/pes/apiv1/loans?first_name=Peter&last_name=Novák&birth_date=1998-0406&person_id=9804066387

Servery uvedené v online Dokumentácii a možnosť testovať jednotlivé EP nie sú súčasťou zadania ani poskytnutých zdrojov.